莫比乌斯反演

2019年6月4日 20:31

一. 多重和式Multiple Sums

1. Iverson bracket艾佛森括号
    i. 当其中条件为真则取1，否则取0
2. Interchanging the order of summation交换求和次序：
    i. $\Sigma(j\in J)\Sigma(k\in K)\ P(j,k) = \Sigma(k\in K)\Sigma(j\in J)P(j,k) = \Sigma j\Sigma k\ P(j,k)*[j\in J]*[\Sigma(k\in K]$
    ii. 其中方括号是相当于布尔逻辑运算，其值为0或1
3. General Distributive Law一般分配律
    i. $\Sigma(j\in J)\Sigma(k\in K)\ Aj*Bk = \Sigma(j\in J)\ Aj * \Sigma(k\in K)\ Bk$
    ii. $[1<j<k<N] = [1<j<N][j<k<N] = [1<k<N][1<j<k]$取小于等于时易知亦成立，是关于$[j\in J][k\in K(j)] = [k\in K'][j\in J'(k)]$的推论
4. 换指标（有时可能顺便换了顺序）
    i. $\Sigma(n|m)\ An$（其中n|m表示n整除m即m是n的整数倍）
    $=\Sigma(n|m)\ Am/n$ （$= \Sigma(m=kn)\ An = \Sigma(m=nk)\ Ak$）
    ii. $\Sigma(k|m)\ \Sigma(m|n)\ a(k,m) = \Sigma(k|n)\ \Sigma(l\mid n/k)\ a(k,kl)$

$=\Sigma j,l,k,m>0[m=kl][n=jm]\ a(k,kl)$	$=\Sigma j,l,k,m>0\ [n/k=ml][n=jk]\ a(k,kl)$
$=\Sigma j,l,k>0\ [n=jkl]\ a(k,kl)$	$=\Sigma l,k,m>0\ [n=kml]\ a(k,kl)$
$=\Sigma t,l,k>0\ [n=tkl]\ a(k,kl)$	$=\Sigma l,k,t>0\ [n=ktl]\ a(k,kl)$

    iii. 即**Σ(小|中) Σ(中|大) a(小,中)=Σ(小|大) Σ(l | 大/小) a(小,l小)**
5. 一个比较综合的例子的两种化简法（Hk是调和级数1/1 + ... + 1/k）
    i. $Sn = \Sigma(1<=j<k<=n)\ 1/(k-j) = \Sigma(1<=j<k<=n)\ 1/(k-j)$

$=\Sigma(1<=j<=n)\ \Sigma(j<k<=n)\ 1/(k-j)$	$=\Sigma(1<=j<t+j<=n)\ 1/t$
$=\Sigma(1<=j<=n)\ \Sigma(j<t+j<=n)\ 1/t$	$=\Sigma(1<=t<=n)\Sigma(1<=\mathbf{j<n-t})\ 1/t$ （t范围可以写<=n-1，但没必要）
$=\Sigma(1<=j<=n)\ \Sigma(0<t<=n-j)\ 1/t$	$=\Sigma(1<=t<=n)\ (n-t)/t$（常数列和）
$=\Sigma(1<=j<=n)\ H(n-j)$	$=\Sigma(1<=t<=n)\ (n/t -1)$
$=\Sigma(1<=n-k<=n)\ Hk$	$=n*\Sigma(1<=t<=n)\ 1/t - n$
$=\Sigma(0<=k<=n-1)\ Hk$	$=n*Hn - n$

    ii. 于是发现调和级数Hn的前n项和是n*Hn-n

二. 欧拉phi函数和莫比乌斯mu函数

1. phi(m)=与m互素的正整数数量（relative prime互素iff [gcd==1]）
    i. phi(1)=1；phi(质数p)=p-1；phi(合数m)<m-1
    ii. 费马小定理的欧拉推广：$n^{phi(m)} \equiv 1 (\mathrm{mod}\ m)$
    iii. phi(质数p^k) = p^k - p^(k-1)
    iv. Σ(d|m) phi(d)=m，应用例：0/12,1/12......11/12化到分母最小，即

0/1,1/2,1/3,2/3,......,1/12,5/12,7/12,11/12中各分母出现次数恰对应12各因子的phi值

v. Σ(d|n) mu(d)/d = phi(n)/n

2. mu(m)的递推式：**Σ(d|m) mu(d) = [m==1]**
   i. 三段式公式：**mu(1)=1；m质因子幂次不都为1时，mu(m)=0；m质因子幂次数均为1时，mu(m)=(-1)^质因子数**
   ii. 举例：mu(质数)=-1，mu(a^2)=0，mu(两不同质数的积)=1

m	1	2	3	4	5	6	7
mu(m)	1	-1	-1	0	-1	1	-1
8	9	10	11	12	13	14	15
0	0	1	-1	0	-1	1	1

3. multiplicative积性：f(1)=1且正整数m1m2互质时f(m1m2)=f(m1)f(m2)
   i. 定理：**g(m)=Σ(d|m) f(d)是积性的 iff f是积性的**（归纳法证）
   ii. 推论：phi函数是积性的，mu函数也是积性的
   iii. 若m1m2不互质也有该性质，称为完全积性函数
4. 除数函数σk(n)：n的各除数的k次幂的和（包括1和n），σ0常记作d

三. 莫比乌斯反演Inversion principle
1. **g(m)=Σ(d|m) f(d)** iff **f(m)=Σ(d|m) mu(d)*g(m/d)**
2. 证明：

已知g(m)=Σ(d|m) f(d)	已知f(m)=Σ(d|m) mu(d)*g(m/d)
则Σ(d|m) mu(d)*g(m/d)	则Σ(d|m) f(d)
=Σ(d|m) mu(m/d)*g(d)	=Σ(d|m) Σ(t|d) mu(t)*g(d/t)
=Σ(d|m) mu(m/d) * Σ(k|d) f(k) （=Σ(k|d) Σ(d|m) mu(m/d) * f(k) =Σ(k|m) Σ(l | m/k) mu(m/ lk) * f(k)）	=Σ(d|m) Σ(t|d) mu(d/t)*g(t)
=Σ(k|m) Σ(d| m/k) mu(m/k /d) * f(k)	=Σ(t|m) Σ(l| m/t) mu(l)*g(t)
=Σ(k|m) Σ(d| m/k) mu(d) * f(k)	=Σ(t|m) [m/t==1] g(t)
=Σ(k|m) [m/k==1] f(k)	=g(t)=g(m)
=f(k)=f(m)	

3. 应用：从 m=Σ(d|m)phi(d) 推出 **phi(m)=Σ(d|m) mu(d)* m/d**
4. 反演另一种形式：**g(m)=Σ(m|d) f(d)** iff **f(m)=Σ(m|d) mu(d/m)*g(d)**

四. 狄利克雷卷积
1. Dirichlet convolution狄利克雷卷积：形如h(m)=Σ(d|m) f(d)* g(m/d)的式子称为~，等价于h(m)=Σ(a*d=m) f(d)* g(a)（a,d,m>0），可记作h=f*g
   i. 满足交换律：f*g=g*f
   ii. 满足结合律：(f*g)*h=f*(g*h)
   iii. 满足加法分配律：f*(g+h)=f*g+f*h

iv. e(n)=[n==1]函数是卷积的单位元，即f*e=e*f=f

v. 莫比乌斯函数与常值一函数即**mu与1(n)=1互为逆元**（因此F=f*1时，可反演出f=mu*F）

vi. 满足积性的“传递性”：若fg均为积性函数，则f*g也为积性函数

vii. 若g是完全积性函数，则h=f*g iff f=h*(mu g)即h(n)=Σ(d|n)f(d)g(n/d) iff f(n)=Σ(d|n)h(d)mu(n/d)g(n/d)

2. 常见狄利克雷卷积
    i. d = 1*1 = Σ(d|n)1 = d(n) = 约数个数
    ii. id = phi*1 = Σ(d|n)phi(d) = id(n) = n本身
    iii. σ = id*1 = Σ(d|n)id(d) = σ(n) = 约数和
    iv. e = mu*1 = Σ(d|n)mu(d) = e(n) = [n==1] = 卷积单位元
    v. f = f*e = Σ(d|n)f(d)[n/d==1] = f(n)
    vi. 1 = mu*d = Σ(d|n)mu(d)*d(n/d) = 1（i的反演）
    vii. phi = mu*id = Σ(d|n)mu(d)*(n/d) = phi(n)（ii的反演）
3. 杜教筛求S(n) = Σ(i=1:n)f(i)，其中f是积性函数
    i. 找另一个积性函数g能使Σ(i=1:n) (g*f)(i) = Σ(i=1:n) Σ(d|i) f(i/d)*g(d) = Σ(d=1:n) g(d) Σ(i=1:n/d) f(i) = Σ(d=1:n) g(d) S(n/d)（向下取整）
    ii. 于是得到**g(1)S(n) = Σ(i=1:n) (g*f)(i) - Σ(d=2:n) g(d)*S(n/d)**
        -
        - 模板&例题

五. 杜教筛模板题

1. 求S(n) = Σ(i=1:n) mu(i)。令g=1(n)=1，由mu*1=e知1*S(n) = Σ(i=1:n) e(i) - Σ(d=2:n) 1*S(n/d) = 1 - Σ(d=2:n) S(n/d)
2. 求S(n) = Σ(i=1:n) phi(i)。令g=1(n)=1，由phi*1=id知1*S(n) = Σ(i=1:n) 1 - Σ(d=2:n) 1*S(n/d) = n*(n+1)/2 - Σ(d=2:n) S(n/d)。算法复杂度是关于d的根号套根号，即O（n^(3/4)）预处理好n^(2/3)的S，可优化为O（n^(2/3)）然而1e9的n，预处理2e6T了，4e6过了

```
int prm[MN],tot;    //prime质数集合及其大小
int mu[MN],phi[MN],smu[MN];
ll sphi[MN];

unordered_map<int,ll>ansphi;
unordered_map<int,int>ansmu;

void init_phi_mu(int N){   //打表[1,N]的mu和phi值
     mu[1] = phi[1] = 1;
     for_(i,2,N){
          if(!phi[i]) prm[tot++] = i, phi[i] = i-1, mu[i] = -1;
          for_(j,0,tot){
               int p = prm[j];
               if(i*p >= MN) break;
               if(i%p==0){
                    phi[i*p] = phi[i]*p;
                    break;}
               phi[i*p] = phi[i]*(p-1);
               mu[i*p] = -mu[i];}}}

ll getphi(int n){   //求phi的前n项和
```

```
        if(n<MN) return sphi[n];
        if(ansphi[n]) return ansphi[n];
        ll ans=(1ll+n)*n/2;
        for__(l,2,n){
            int r=n/(n/l);
            ans-=ll(r-l+1)*getphi(n/l);
            l=r;   //下一次处理r+1分块
        }
        return ansphi[n]=ans;
    }

    int getmu(int n){   //求mu的前n项和
        if(n<MN) return smu[n];
        if(ansmu[n]) return ansmu[n];
        int ans=1;
        for__(l,2,n){
            int r=n/(n/l);
            ans-=(r-l+1)*getmu(n/l);
            l=r;   //下一次处理r+1分块
        }
        return ansmu[n]=ans;
    }
```

六. 莫比乌斯模板题Σ(i=1:a) Σ(j=1:b) [gcd(i,j)==k]

1. 易知Σ(i=1:a/k) Σ(j:1~b/k) [gcd(i,j)==1]可与Σ(i=1:a) Σ(j=1:b) [gcd(i,j)==k]各有序对(i,j)一一对应起来，所以值相同
2. 令f(n)=Σ(i=1:a/k) Σ(j:1~b/k) [gcd(i,j)==n]，则f(1)即为所求
3. 令F(n)=Σ(n|d) f(d) = Σ(n|d) Σ(i=1:a/k) Σ(j=1:b/k) [gcd(i,j)==d]=该范围内gcd为n的倍数的有序对(i,j)数量=a/k/n * b/k/n
4. 反演出f(n)=Σ(n|d) mu(d/n)*F(d)
5. 所以答案=f(1)=Σ(d=1:min{a/k,b/k}) mu(d)* a/k/d * b/k/d
6. 注意ll，注意给除法打括号

```
    void init_mu(int N){        //打表[1,N]的mu值
        mu[1] = 1;
        for__(i,2,N){
            if(!npr[i])
                prm.push_back(i),
                mu[i] = -1;   //mu(质数)=-1
            for(auto p:prm){
                if(i*p > N) break;  //爆表了
                npr[i*p] = 1;
                if(i%p==0){
                    mu[i*p] = 0;
                    break; //质因子p在分解中幂次>1时mu=0
                }else  //质因子p幂次为1
                    mu[i*p] = -mu[i];
            }
        }
    }
        init_mu();
        cin>>a>>b>>k; a/=k; b/=k;
        for__(d,1,min(a,b)) ans+= mu[d]* (a/d) * (b/d);
        cout<<ans<<'\n';
```

七. 积性函数推F=id^k函数和mu函数的狄利克雷卷积 牛客18949

1. 从小到大用线性筛递推

2. 当互质时，可以直接利用积性做乘法得到
3. 当i%prime[j]==0时，可以利用积性，对F(不是prime[j]^?)乘以F(prime[j]^?)
4. 而对F(prime[j]^k)展开id^k和mu函数，发现随k递增也只需累乘

八. 银川ICPC金牌题D：Σ_i1=1^m Σ_in=1^m[gcd(i1,...,in)==d](i1...in)^k

1. https://nanti.jisuanke.com/t/42384
2. 提出d，**d^kn** Σ_i1=1^**m/d** Σ_in=1^**m/d**[gcd(i1,...,in)**==1**](i1...in)^k
3. 反演 d^kn Σ_i1=1^m/d Σ_in=1^m/d (**Σj|gcd(i1,...,in) mu(j)**) (i1...in)^k
4. 换求和顺序（由gcd范围1:m/d知j=1:m/d，i=1:m/dj等价原式，记得i改i'j）
   d^kn **Σ_j=1^m/d mu(j)** Σ_i1=1^**m/dj** Σ_in=1^**m/dj** (i1**j**...in**j**)^k
5. 无关项提前（n个i'乘的j的k次方可提，ix之间互相无关可先各自求和完再乘）
   d^kn Σ_j=1^m/d mu(j) **j^kn** Σ_i1=1^m/dj i1^k Σ_in=1^m/dj in^k
6. 化简得d^kn Σ_j=1^m/d mu(j) j^kn (Σ_i=1^m/dj i^k)^n
7. 即，预处理好1:m/d的mu(i)，j^k及其前缀和，n较大，需要边读入边降幂，数据较弱，没有出现欧拉降幂是否大于phi的那个特判

```
const int P = 59964251;
ll euler(ll n){
	ll ans=n, ed=sqrt(n);
	for(int i=2; i<=ed; ++i)
		if(n%i==0){
			ans-=ans/i;
			while(n%i==0) n/=i;}
	if(n>1) ans-=ans/n;
return ans;}

const int phiP = euler(P);
const int MN = 1e5 + 5;

bitset<MN>npr;
int mu[MN];
vector<int>prm;
void init_mu(int N){		//打表[1,N]的mu值
	mu[1] = 1;
	for(int i=2; i<=N; ++i){
		if(!npr[i])
			prm.push_back(i),
			mu[i] = -1;	//mu(质数)=-1
		for(auto p : prm){
			if(i*p > N) break;	//爆表了
			npr[i*p] = 1;
			if(i%p == 0){
				mu[i*p] = 0;
				break; //质因子p在分解中幂次>1时mu=0
			}else	//质因子p幂次为1
				mu[i*p] = -mu[i];
		}
	}
}

int qpow(ll a,int b){
	ll rt=1;
	for(; b; b>>=1, (a*=a)%=P)
		if(b&1) (rt*=a)%=P;
return rt;}

char str[MN];
ll sum[MN];
ll n; int m,d,k;
inline int gao(int j){
```

```
    return qpow(j,n*k%phiP+phiP)%P*qpow(sum[m/j],n+phiP)%P;
}
int main(int argc, char** argv){
    init_mu(1e5+2);
    int _; scanf("%d",&_);
    while(_--){
        scanf("%s",str);
        int len=strlen(str);
        n=0;
        for(int i=0; i<len; ++i)
            n=(n*10+str[i]-'0')%phiP;

        scanf("%d%d%d",&m,&d,&k);
        m/=d;
        for(int j=1; j<=m; ++j)
            sum[j]=(sum[j-1]+qpow(j,k))%P;

        ll ans=0;
        for(int j=1; j<=m; ++j)
            (ans+=ll(mu[j])*gao(j)+P)%=P;    //mu可能为负，需要+P
        (ans*=qpow(d,n*k%phiP+phiP))%=P;
        printf("%lld\n",ans);
    }
    return 0;
}
```